Δεύτερη Σειρά ασκήσεων
Ημερομηνία Παράδοσης: 14 Μαΐου 2015, 12 μ.μ.

Οι παρακάτω ασκήσεις πρέπει να παραδοθούν μέχρι τις 14/5/2015 12μ.μ. . Στην υλοποίηση των κλάσεων σας δεν θα πρέπει να έχετε public μεταβλητές. Βαθμοί θα αφαιρεθούν για προγράμματα που δεν είναι καλά γραμμένα, δηλαδή δεν είναι σωστά στοιχισμένα ή δεν έχουν καλά επιλεγμένα ονόματα μεταβλητών ώστε να διαβάζονται εύκολα.

Άσκηση 1

Στην πρώτη σειρά ασκήσεων υλοποιήσατε τον Αφηρημένο Τύπο Δεδομένων **Ουρά (Queue)**, χρησιμοποιώντας ένα πίνακα σταθερού μεγέθους. Σε αυτή την άσκηση θα υλοποιήσετε μια Ουρά χωρίς περιορισμό στο μέγεθος. Η ιδέα της υλοποίησης είναι παρόμοια με την υλοποίηση μιας στοίβας με συνδεμένα στοιχεία που είδαμε στην τάξη. Η διαφορά είναι ότι, όπως και στην προηγούμενη άσκηση, εκτός από μία αναφορά που δείχνει στην κορυφή της ουράς (**head**) και μία αναφορά που δείχνει στο τέλος (**tail**) της ουράς. Τα στοιχεία προστίθενται στο τέλος της ουράς.

Η Ουρά θα κρατάει αντικείμενα τύπου **Person**, που κρατάνε πληροφορία για ένα άτομο. Το κάθε άτομο θα έχει ένα όνομα (**name**) και ένα αριθμό μητρώου (**ΑΜ**). Η βάση της σας δίνεται στην σελίδα του μαθήματος, πρέπει να την επεκτείνετε με επιπλέον μεθόδους.

Κατασκευάσετε την κλάση **Queue** που υλοποιεί μια ουρά που αποθηκεύει αντικείμενα Person. Η κλάση θα πρέπει να έχει τις εξής μεθόδους:

- Ένα **constructor** χωρίς ορίσματα.
- Ένα **copy constructor** που παίρνει σαν όρισμα ένα αντικείμενο Queue και δημιουργεί ένα **βαθύ** αντίγραφο. Χρησιμοποιήστε φωλιασμένους copy constructors για να το υλοποιήσετε.
- Μία μέθοδο **insert** η οποία παίρνει σαν όρισμα ένα αντικείμενο Person και τον προσθέτει στο τέλος της ουράς.
- Μία μέθοδο **remove** η οποία αφαιρεί και επιστρέφει το αντικείμενο Person στην κορυφή της ουράς, ή null αν είναι άδεια η ουρά.
- Την μέθοδο **toString** η οποία επιστρέφει ένα αλφαριθμητικό με τα στοιχεία της ουράς σε FIFO σειρά, χωρισμένα με κενό. Η toString() θα υλοποιηθεί καλώντας φωλιασμένες toString(). Η μέθοδος toString() για την Person θα πρέπει να επιστρέφει το όνομα και το ΑΜ μέσα σε μία παρένθεση χωρισμένα με κόμμα.
- Την μέθοδο **equals** η οποία ελέγχει αν η ουρά είναι ίδια με κάποια άλλη. Ισότητα σημαίνει ότι οι δύο ουρές περιέχουν ακριβώς τα ίδια στοιχεία με ακριβώς την ίδια σειρά. Χρησιμοποιήστε φωλιασμένες equals για να την υλοποιήσετε.
- Την μέθοδο **merge** η οποία παίρνει σαν όρισμα ένα αντικείμενο Queue, και συγχωνεύει την ουρά που δίνεται σαν όρισμα, στην ουρά που καλεί την μέθοδο.
- Την μέθοδο **isEmpty** η οποία επιστρέφει true ή false ανάλογα αν η ουρά είναι άδεια ή όχι.
- Μια μέθοδο **size** που επιστρέφει τον αριθμό στοιχείων στην ουρά.

Ορίστε μια μέθοδο main για να τεστάρετε την κλάση σας. Για να σας βαθμολογήσουμε θα φτιάξουμε μια κλάση **QueueTestPerson** που θα χρησιμοποιεί την Queue που θα παραδώσετε. Ένα παράδειγμα δίνεται στην σελίδα του μαθήματος. Θα παραδώσετε μόνο το αρχείο Queue.java.

Υποδείξεις

- Για να πάρετε όλους τους βαθμούς θα πρέπει να φροντίσετε και τις οριακές καταστάσεις (δηλαδή τις περιπτώσεις που μπορεί να δημιουργηθεί κάποιο λάθος).
- Ορίστε κάποια επιπλέον κλάση αν την χρειάζεστε.

Άσκηση 2

Στην άσκηση αυτή θα προσομοιώσετε την λειτουργία ενός κοινωνικού δικτύου. Το κοινωνικό δίκτυο έχει χρήστες (users), οι οποίοι έχουν φιλίες μεταξύ τους. Ο καθένας έχει ένα τοίχο (wall) στον οποίο εμφανίζονται σε αντίστροφη χρονολογική σειρά τα μηνύματα (posts) που κάνουν ο χρήστης και οι φίλοι του. Οι χρήστες μπορούν επίσης να κάνουν σχόλια στα posts που εμφανίζονται στον τοίχο τους.

Το πρόγραμμα σας θα πρέπει να έχει τρεις βασικές κλάσεις: **User**, **Post**, **Wall**.

Η κλάση **Post** κρατάει πληροφορίες για ένα post. Συγκεκριμένα χρειάζεται να κρατάει ένα String με το κείμενο του Post, ένα αντικείμενο User που αντιστοιχεί στον συγγραφέα του Post, και ένα ArrayList που κρατάει Post αντικείμενα τα οποία είναι τα σχόλια για αυτό το post. Ο **constructor** της κλάσης παίρνει σαν όρισμα τον συγγραφέα και το κείμενο του post. Η μέθοδος **toString**, δημιουργεί ένα String που περιέχει το String του χρήστη, το κείμενο του post και, εφόσον υπάρχουν σχόλια, τα Strings για όλα τα σχόλια που εμφανίζονται για το post. Για παράδειγμα παρακάτω είναι το String για δύο post του χρήστη Alice, ένα με σχόλια, και ένα χωρίς.

```
Alice: Earthquake in Nepal!
Comments:
        Bob: This is terrible!
        Charlie: Devastating!
```

```
Alice: Travelling home today!
```

Η κλάση έχει επίσης μια μέθοδο **addComment** που παίρνει σαν όρισμα ένα post και το προσθέτει στην λίστα με τα σχόλια.

Η κλάση **Wall** κρατάει πληροφορίες για τον τοίχο ενός χρήστη. Συγκεκριμένα κρατάει ένα αντικείμενο User που είναι ο ιδιοκτήτης του τοίχου, και ένα ArrayList που κρατάει αντικείμενα τύπου Post που είναι τα posts που εμφανίζονται στον τοίχο του χρήστη. Ο **constructor** παίρνει σαν όρισμα τον ιδιοκτήτη του τοίχου. Έχει μια μέθοδο **addPost** που παίρνει σαν όρισμα ένα αντικείμενο Post και το προσθέτει στην λίστα, και μια μέθοδο **addComment** που παίρνει σαν όρισμα ένα ακέραιο i και ένα Post p και προσθέτει το p στα σχόλια του i-οστού post της λίστας (εφόσον η λίστα έχει τουλάχιστον i posts).
Επίσης υπάρχει και η μέθοδος **display** η οποία εμφανίζει τον τοίχο: Την πληροφορία για τον ιδιοκτήτη του τοίχου, και τα posts με αντίστροφη σειρά με την οποία εμφανίζονται (το τελευταίο post εμφανίζεται πρώτο), μαζί με το νούμερο τους. Για παράδειγμα παρακάτω είναι μια ενδεικτική έξοδος για τον τοίχο του χρήση Alice, ο οποίος έχει δύο post ένα με σχόλια και ένα χωρίς:

```
Wall for user Alice:

1. Alice: I am travelling home today!

0. Alice: Earthquake in Nepal!
Comments:
        Bob: This is terrible!
        Charlie: Devastating!
```

Η κλάση **User** κρατάει πληροφορίες για ένα χρήστη. Συγκεκριμένα κρατάει το όνομα του χρήστη, ένα αντικείμενο Wall που αντιστοιχεί στον τοίχο του χρήστη, και ένα ArrayList που κρατάει τους φίλους του χρήστη. Ο **constructor** παίρνει όρισμα το όνομα και η toString επίσης επιστρέφει το όνομα. Η κλάση έχει μία μέθοδο **befriend** που παίρνει σαν όρισμα έναν άλλον χρήστη και δημιουργεί μια φιλία μεταξύ των δύο χρηστών (ο ένας εμφανίζεται στην λίστα του άλλου). Η μέθοδος **post** παίρνει σαν όρισμα ένα String και κάνει ένα post με αυτό το κείμενο, δηλαδή, προσθέτει το post στον τοίχο του χρήστη καθώς και στον τοίχο όλων των φίλων του χρήστη. Η μέθοδος comment παίρνει σαν όρισμα ένα ακέραιο i και ένα String s, και προσθέτει ένα σχόλιο στο i-οστο post στον τοίχο του χρήστη με αυτό το κείμενο.

Η κλάση User έχει και μία μέθοδο **visitWall** η οποία υλοποιεί την αλληλεπίδραση του χρήστη με τον τοίχο. Συγκεκριμένα, εμφανίζει τον τοίχο του χρήστη, και μετά τον ρωτάει αν θέλει να ποστάρει (post - p), να σχολιάσει (comment - c), ή να φύγει (exit - e). Αν επιλέξει να ποστάρει, δίνει το κείμενο για το post και δημιουργεί ένα νέο post, αν επιλέξει να σχολιάσει του ζητείται να εισάγει το νούμερο του post που θέλει να σχολιάσει και το κείμενο του σχολίου, και αν επιλέξει να φύγει βγαίνει από την μέθοδο.

Τέλος θα υλοποιήσετε μια κλάση **SocialNework**, η οποία θα περιέχει την **main** και θα κάνει προσομοίωση της λειτουργίας του κοινωνικού δικτύου. Ενδεικτικά προτείνεται να κάνετε ένα κοινωνικό δίκτυο από 5 χρήστες που μπορείτε να συνδέσετε με διάφορους τρόπους μεταξύ τους. Μέσα σε ένα βρόγχο οι χρήστες θα επισκέπτονται με την σειρά τον τοίχο τους και θα κάνουν καινούρια posts ή σχολιασμό στα υπάρχοντα posts. Το πρόγραμμα θα σταματάει όταν το επιθυμείτε.

Μπορείτε να κάνετε την προσομοίωση σας όσο περίπλοκη θέλετε. Εμείς θα κάνουμε μια δικιά μας υλοποίηση της SocialNetwork που θα χρησιμοποιήσουμε για την βαθμολόγηση του προγράμματος σας.

Μπορείτε να χρησιμοποιήσετε επιπλέον μεθόδους ή πεδία αν το χρειάζεστε.